IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS), Hamburg, Germany, pp. 3144-3151, 2015

IRON: A Fast Interest Point Descriptor for Robust NDT-Map Matching and its Application to Robot Localization*

Thomas Schmiedel, Erik Einhorn and Horst-Michael Gross[1]

***Abstract*— This work introduces the IRON keypoint detector and the IRON descriptor which enable high-speed and high-accuracy alignment of 3D depth maps. Instead of using raw point values for storing 3D-scenes, all algorithms were designed to operate on *Normal Distribution Transforms* (NDT), since NDT-maps provide a highly memory-efficient representation of depth data. By taking into account surface curvature and object shape within NDT-maps, patches with strong surface variability can be recognized and described precisely. In this paper, the whole feature extraction process, as well as descriptor matching, outlier detection, and the final transform calculation between NDT-maps is elaborated. The presented technique is particularly insensitive to an initial offset between both maps, has a high robustness, and it achieves more than 75 NDT-map alignments per second (including complete memory allocation each time as well) in two large publicly available depth datasets while using only a single core of a modern Intel i7 CPU. Even though the main focus of this work was placed on the proposed IRON registration algorithm, two specific applications of this NDT-matching approach are outlined in the second part, namely robot pose tracking and NDT-one-shot localization within densely furnished domestic environments.**

I. Introduction

Localizing itself within the environment is – besides map building and path planning – one of the most important capabilities of a mobile robot. It therefore has been extensively studied, and the Adaptive Monte Carlo Localization (AMCL[1])[1] in combination with 2D laser range finders evolved to the de facto standard. However, in real-world applications there are still accuracy issues that arise in certain locations, such as densely furnished domestic environments. In these places, localization can greatly benefit from 3D information obtained by depth cameras, which provide much more information about the structure of local surroundings. However, since 3D depth sensors produce a huge amount of data, an appropriate representation is needed for processing this quantity efficiently. Therefore, we use Normal-Distribution-Transform (NDT) maps as they provide a compact representation of 3D structure by storing the probability of observing a surface at a certain point in space [2]. As shown in [3], NDT-maps achieve a significantly higher accuracy than voxel maps when the same cell resolution is used.

In this paper, we present a novel **I**nterest point descriptor for **RO**bust **N**DT-map matching (IRON) that allows to recognize and describe normal distribution patterns within one map, to find their counterparts inside a different map, and to match them accordingly. This enables fast and accurate NDT-map registration (see Fig. 1), and we show that our technique is therefore well suited for robot localization using 3D sensors. Basically, two major variants of robot localization are distinguished: *global localization* and *pose tracking*. Global localization is considered the task of finding the pose of the robot without any prior knowledge except a given global map. Once this pose has been found, localization becomes the objective of maintaining an estimate of the pose, which is referred to as pose tracking [4]. To perform global localization, the robot would normally explore its environment by driving around. This, however, poses the risk of moving into areas where it is not allowed to go. Therefore, we introduce a challenging variant of global localization here that we name *NDT-one-shot localization*. It is the task of finding the robot's pose using only a single sensor reading which is obtained while the robot is standing still or rotates in place.

This paper is organized as follows: The next section outlines the current state-of-the-art in terms of 3D point cloud registration, NDT-map registration, and localization based on NDT-maps. In sections III-V, we describe our keypoint detector, the descriptor, and the map matching technique in detail. To show potential applications, a pose tracking and a one-shot localization method are described in section VI. In section VII, we show several results and analyses of the presented approaches and finally conclude with an outlook for future work.

II. State-of-the-Art in 3D-map registration

The task of 3D point cloud registration has already been addressed in multiple publications, such as the ones about

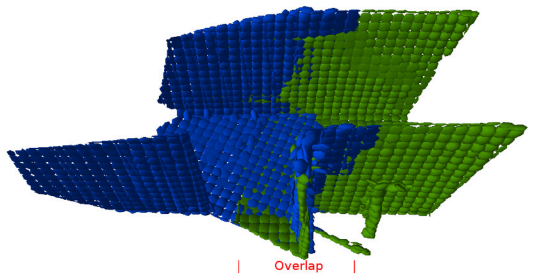


Fig. 1: Two NDT-maps (blue and green) from a corridor scene were registered using the IRON-algorithm. Note that only a small overlap is available for matching.

*This work has received funding from the German Federal Ministry of Education and Research as part of the ROREAS project under grant agreement no. 16SV6133.

[1]T. Schmiedel, E. Einhorn, and H.-M. Gross are with Neuroinformatics and Cognitive Robotics Lab, Ilmenau University of Technology, 98694 Ilmenau, Germany. thomas.schmiedel@tu-ilmenau.de

[1]http://wiki.ros.org/amcl

NARF-, SURE-, ISS-, FPHF-, 3DSC- and SHOT-features [5][6][7][8][9][10]. However, a major reason for the use of NDT-maps is their efficient resource utilization while at the same time much of the structural information available from the scene is preserved[2][3]. This special 3D representation, in turn, needs matching techniques that can handle the contained multivariate normal distributions. It was suggested by Stoyanov et al., to use a strategy similar to *Iterative Closest Point* (ICP) for NDT-map registration, with the distance metric chosen to be the L_2-norm between Gaussian Mixture Models [11]. This approach yields good results, but was mainly tested with Velodyne LIDAR devices in large scale environments. Since the Velodyne has a viewing range of $360°$, successive NDT-maps have a wide overlap available for matching.
Saarinen et al. successfully combined this ICP-based matching principle with the well established Monte Carlo Localization (MCL) to form a localization technique that was used within warehouses [12].

Our work, however, is focussed on robots for home environments, equipped with sensors (Microsoft Kinect, Asus Xtion) with a much smaller viewing angle and shorter visibility range, respectively. Also, due to our robot's small dimensions [13][14], major changes in the sensor's orientation are to be expected within short time frames. This implies, that a registration algorithm – processing depth data from small scale environments – must be particularly insensitive to a large initial displacement and rotation between two successively captured 3D maps. A high processing speed is especially helpful as well, since small rooms and the Kinect sensor's limited viewing range require short update intervals. Therefore, it was of utmost importance for us to design a fast registration system while maintaining state-of-the-art registration accuracy and a high robustness.
Apart from the used data structure (point clouds vs. NDT-maps), IRON has some major conceptual differences compared to current point-based methods. Most of them (ISS, SURE, SHOT, FPFH, 3DSC to some extend) build a local frame of reference around each keypoint in order to find distinct feature descriptors. This, in turn, requires the direction (positive or negative sign) of surface normal vectors to be known beforehand. A common approach is to flip all normals into the direction of the depth sensor. If, however, a depth scene was captured from two different locations, normal directions will be different as well; and so will the descriptors they form. To avoid this potential source of error, IRON does not rely on the correct sign of surface normals, it will give equal results no matter where the scene was captured from.
A similarity to SURE can be seen in the way salient regions are identified. Both IRON and SURE utilize entropy computations within their keypoint detectors [6][15], however, the way this is done is quite different. SURE searches for spikes in the local surface entropy, while, for efficiency reasons, IRON takes a part of the descriptor itself and computes the normalized histogram entropy for this part (see Sec. III), which is then used to mark whole patches within the NDT-map as keypoints.
Regarding one-shot localization, research has mainly focussed on vision-based strategies [16] which, however, are computationally expensive. NDT-maps might provide a fast alternative for this challenge.

III. Keypoint Detector

In order to align two NDT-maps, we propose a technique that detects local regions of salient surface curvature and characterizes those areas with high distinction. Once all keypoints and their corresponding descriptors have been created separately for each map, they will be matched and subsequently searched for outliers.

NDT-maps are composed of spatially disjoint NDT-cells, where every NDT-cell contains a three dimensional multivariate normal distribution $\mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ parameterized by its mean vector $\boldsymbol{\mu}$ and covariance matrix $\boldsymbol{\Sigma}$, representing the probability $p(\mathbf{x})$ to observe a surface for any given point $\mathbf{x}$ within that cell. Therefore, if the distribution is flat but not needle-shaped, the eigenvector corresponding to the smallest of the three eigenvalues of $\boldsymbol{\Sigma}$ gives a good approximation of the local surface normal.

The proposed keypoint detector computes these surface normals and ignores cells with a high ratio $\lambda_{min}/\lambda_{med}$ between the smallest eigenvalue and the medium eigenvalue, to reject spherically shaped NDT-components. Additionally, cells are only permitted if they have a high ratio $\lambda_{med}/\lambda_{max}$, to suppress needle-shaped distributions.

After the surface normals for every suitable NDT-cell have been computed, the keypoint detector marks those cells, whose surrounding volume within a radius ϵ implies a high amount of structural information, as expressed by strongly varying surface normal orientations. The intention is to speed up the following descriptor matching process by only focussing on parts of the map that carry most of its curvature information. At the same time, this favors keypoint descriptors that represent very distinct features within the scene and reduces their overall ambiguity, which, in turn, improves the matching quality later on. Any cell at the center of a flat wall, for instance, will be almost impossible to detect inside other NDT-maps, as there is no way to reliably differentiate between flat surfaces without color information.

To detect keypoints, the following algorithm is repeated for each NDT-cell N_i within the NDT-map. This currently processed cell N_i is called *base cell* in the following:

- Create a matrix $\mathbf{A}$ with m rows, n columns, and initialize all elements to zero.
- Find all neighboring NDT-components $N_k \in K$ within radius ϵ around the base cell N_i. This can be efficiently implemented by means of a k-d-tree data structure.
- For each neighbor N_k, compute the Euclidean distance $d = ||\boldsymbol{\mu}_k - \boldsymbol{\mu}_i||_2$ between its mean vector $\boldsymbol{\mu}_k$ and $\boldsymbol{\mu}_i$ of the base cell and determine the smallest angle δ among the normal vectors of N_k and N_i, as outlined in Fig. 2.
- For each pair (d, δ), increment the matrix element $a_{p,q}$ of matrix $\mathbf{A}$. The indices p, q are computed as given in Eq.(1) and Eq.(2).

- After all $N_k \in K$ have been processed, every single row p, as well as the matrix $\mathbf{A}$ itself, must be normalized.
- Finally, the entropy H is calculated and normalized as shown in Eq.(3). If the result exceeds a certain threshold, the base cell is classified as *keypoint* accordingly.

The reasoning behind this procedure is described in detail below. Matrix $\mathbf{A}$ can be seen as a two-dimensional histogram that separates the sphere with radius ϵ around the base cell into spherical shells, each one identified by the row number $p, (1 \leq p \leq m)$. All neighboring cells with a similar distance d to the base will therefore reside in the same matrix row. The index $q, (1 \leq q \leq n)$, however, is a discrete measure of the angle δ between the normal vectors of N_i and N_k.

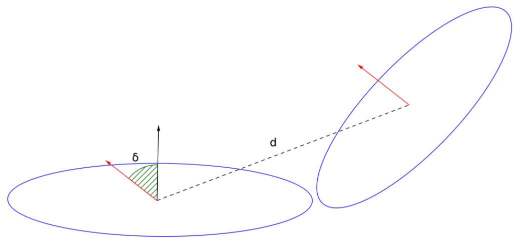


Fig. 2: Computation of a (d, δ)-tuple between the base NDT-cell and a neighbor cell $N_k \in K$

It is important to note that the sign of an eigenvector is random. Hence, from a local point of view, it cannot efficiently be determined whether the computed NDT-cell's normal vector is pointing up or down. Also, for efficiency reasons, we do not create a local reference frame for each descriptor, like it is done in most point-based methods (e.g. FPFH [8]). The value δ is therefore defined as the smallest possible angle between two NDT-normals, thereby limiting its range to: $0 \leq \delta \leq 0.5\pi$ rad. In order to assign every (d, δ)-pair to a certain location inside matrix $\mathbf{A}$, a discretization step follows:

$$p = 1 + \lfloor m \cdot d^2/(\epsilon^2 + c) \rfloor, c = 0.000001 \qquad (1)$$

$$q = 1 + \lfloor n \cdot \delta/(0.5\pi + c) \rfloor, c = 0.000001 \qquad (2)$$

Variable c only prevents overflowing the matrix indices and should be as small as possible. Using the squared distance d^2 in Eq. (1) for discretization gives spherical shells a larger volume if they are close to the base. This ensures that the corresponding first matrix rows will not be sparsely populated. After all $N_k \in K$ neighbors have been processed, each row p is divided, firstly, by the amount of (d, δ)-tuples it contains and, secondly, by the number of non-empty rows inside $\mathbf{A}$, thereby confining all matrix elements to the range $0 \leq a_{p,q} \leq 1$. At this point, $\mathbf{A}$ gives a good impression of the local surface curvature around its base. The following example $(m = 3, n = 4)$ shows its appearance, assuming N_i is located at a flat wall:

$$\mathbf{A} = \begin{Bmatrix} 0.33 & 0 & 0 & 0 \\ 0.33 & 0 & 0 & 0 \\ 0.33 & 0 & 0 & 0 \end{Bmatrix} \begin{array}{l} \leftarrow \text{distance range 1} \\ \leftarrow \text{distance range 2} \\ \leftarrow \text{distance range 3} \end{array}$$

As expected, all angles are close to zero. However, for base cells at curved surfaces, $\mathbf{A}$ will be filled more evenly. To distinguish between those cases, we compute the entropy over all matrix elements and presume, they are different symbols $z_{p,q}$ from a memoryless information source Z with a certain probability of occurrence denoted by entry $a_{p,q}$.

$$H = -\sum_{p=1}^{m}\sum_{q=1}^{n} s_{p,q} \text{ with } s_{p,q} = \begin{cases} 0 & : a_{p,q} = 0 \\ a_{p,q}\log_2 a_{p,q} & : a_{p,q} > 0 \end{cases} \qquad (3)$$

$$H_{max} = \log_2(mn) \qquad (4)$$

The maximally possible entropy H_{max} is limited by the number of places inside matrix $\mathbf{A}$. This way, the ratio H/H_{max} becomes 1 if all (d, δ)-tuples are uniformly distributed. Whenever the normalized entropy for a given NDT-base exceeds a certain threshold, the cell is classified as *keypoint* accordingly. An example is shown in Fig. 3.

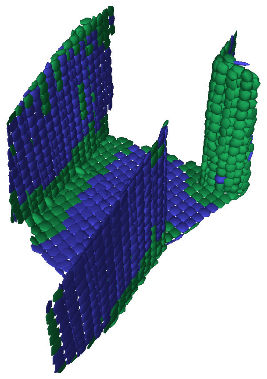


Fig. 3: NDT-map (blue) of a corridor scene. Cells, identified as keypoints, are highlighted in green. The radius ϵ was set to 0.2 m. It is clearly visible that NDT-cells inside curved surface areas are more likely to be classified as keypoints.

IV. Keypoint Descriptor

After the keypoints have been extracted, they need to be described precisely in order to identify corresponding keypoints within different NDT-maps. One can show that the previously constructed matrix $\mathbf{A}$ is invariant to rotation, as it encodes its close environment in terms of the centrally located base cell. This is an important prerequisite for comparing maps later on that have been captured from different locations. Owing to this property, matrix $\mathbf{A}$ already forms the first part of the final keypoint descriptor. This is also a computationally efficient strategy, since existing data can be reused without modification. In order to store information about the *shape* of the scene around the base cell N_i, a different matrix $\mathbf{S}$ is generated. $\mathbf{A}$ and $\mathbf{S}$ are formed in a similar fashion, moreover their dimensions are identical. However, while δ_A is determined as smallest positive angle between the surface normals, δ_S will rather be computed using the normal vector of the base and the direction vector from $\boldsymbol{\mu}_i$ to $\boldsymbol{\mu}_k$, where $\boldsymbol{\mu}_k$ denotes the mean vector of a neighboring distribution N_k within radius ϵ around N_i, as shown in Fig. 4.

Again, each $N_k \in K$ around N_i will produce a (d, δ)-pair that becomes discretized accordingly and finally increments matrix element $s_{p,q}$. Afterwards, matrix $\mathbf{S}$ is normalized the

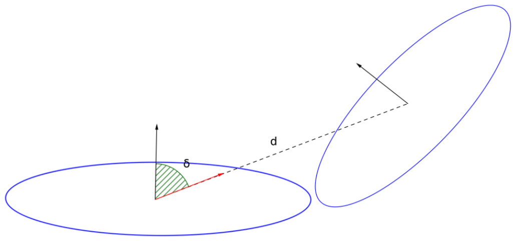


Fig. 4: Computation of a (d, δ)-tuple to be inserted into shape matrix S

same way it was previously done with matrix $\mathbf{A}$, thereby giving all distance rows equal weight and ensuring all matrix elements sum up to 1. The complete descriptor $\mathbf{D}$ now encodes the *local surface curvature* and the *shape* of the scene around a keypoint as well as the base-cell's mean vector $\boldsymbol{\mu}_i$ and the number of incorporated neighbors k: $\mathbf{D} = \{\mathbf{A}, \mathbf{S}, \boldsymbol{\mu}_i, k\}$. It should be noted that whenever the *location of a descriptor* is mentioned below, this refers to the mean vector $\boldsymbol{\mu}_i$ of its base NDT-cell N_i.

V. Descriptor Matching and Outlier Detection

Matching of the descriptor lists from two different NDT-maps is realized via approximate nearest neighbor search inside the $(m \times 2n)$-dimensional feature space using the L_2-norm metric. In particular, the matching algorithm computes crosswise distances between all descriptors of both maps and finally selects pairs that fit best (see [7], Sec. 3.2. for a detailed explanation). The result is a set $M = \{(D_{a1}, D_{b1}), (D_{a2}, D_{b2}), ..., (D_{an}, D_{bn})\}$ that combines descriptors from two different maps as tuples, which all separately form a hypothesis about how both NDT-maps may fit together. Due to similarities between different parts of the scene, it is likely that some of the matches inside M are erroneous. Actually, depending on the chosen NDT-cell size, the conversion from point clouds into NDT-maps inevitably leads to low-pass filtering the data. This, in turn, puts a limit on the expressiveness a single descriptor can have. Hence, before the final map-transform can be computed, an outlier detection step is mandatory. For this purpose, we implemented the RANSAC algorithm using six degrees of freedom $(x, y, z, roll, pitch, yaw)$. After a fixed number of RANSAC-iterations, the result is an outlier-free descriptor set, which can be used to compute the correct transform between both NDT-maps, e.g. via Gauss-Newton optimization. Examples are shown in Fig. 5 and 6.

VI. Application to Localization

Due to the keypoint detection stage that filters unnecessary NDT-cells, recycling of matrix $\mathbf{A}$, k-d-tree data structures and the possibility to cache previously computed descriptors, the proposed algorithm has a high processing speed (Sec. VII-C). Therefore, it is well suited for a Monte Carlo Localization approach with short update intervals.

A. IRON-MCL

The Monte Carlo Localization (MCL) is a technique to track the pose of a mobile robot while it moves and

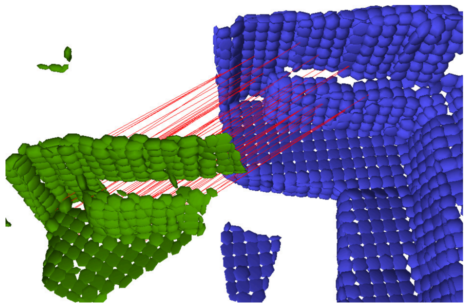


Fig. 5: Corresponding descriptors from both NDT-maps have been connected by red lines. The 3D scene shows parts of a couch.

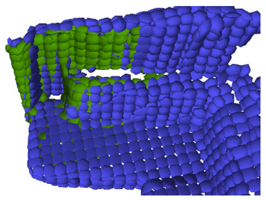


Fig. 6: Based on the outlier-free tuple set M, the final map transform was obtained as best fit in the least-squares sense.

continuously senses its environment [17]. It is realized by means of recursive Bayesian estimation and uses a set of particles $S_t = \left\{(x_t^{[i]}, w_t^{[i]})|i = 1, 2, ..., N\right\}$ and a reference map of the scene to assign each pose-hypothesis x_t an importance weight w_t based on sensor measurements; in case of the suggested IRON-MCL, particle verification is implemented through feature-based NDT-map matching.
As a first step, the global reference map has to be created. This can be achieved by manually navigating the robot inside the area of interest and at the same time running an NDT-SLAM-framework [18] that gradually inserts new NDT-maps and odometry data and finally delivers a global NDT-map of the environment. The actual pose tracking is consistent with the original (A)MCL algorithm [1]. We start by spreading particles normally distributed at the presumed start position of the robot within the map. The following steps are repeated afterwards:

- The robot moves a specific distance or rotates a certain angle, after which an update step is triggered.
- All particles are then handled by the motion model to take into account the robot motion since last update.
- At this point, a fresh NDT-map is captured and its keypoint descriptors are computed.
- With the estimated posterior distribution of the particles, it is possible to guess the current pose of the robot, thereby moving the new NDT-map to its assumed position in world coordinates.
- Now, precomputed descriptors from the global NDT-map are selected if they are within a certain radius around the current local map.
- Descriptor lists from both NDT-maps are then matched and subsequently searched for outliers.
- The importance weight $w_t^{[i]}$ of each particle is determined by placing the current NDT-map according to

pose hypothesis $x_t^{[i]}$ and computing a fitness score c (Eq.5); this value can be considered an approximate probability from the observation model and is assigned to $w_t^{[i]}$ accordingly.

Score c for two NDT-maps is found by iterating all descriptor tuples $(D_{ai}, D_{bi}) \in M$ and computing the Euclidean distance between them. Parameter ρ affects, how eagerly the MCL-particles converge:

$$c = \frac{1}{|M|} \sum_{i=1}^{|M|} \exp -\rho ||\mu_{ai} - \mu_{bi}||_2 \tag{5}$$

B. One-Shot-Localization

A special challenge emerges if the current robot pose is completely unknown. This can simply be the case after switching the machine on, or due to severe localization errors. However, as the robot's position is uncertain, obstacle avoidance is strongly impaired. Therefore, the robot shouldn't have to *move* in order to identify its real location. We propose a technique, based on the previously explained feature-based NDT-matching that can determine the sensor pose inside a given global NDT-map based on a single NDT-snapshot. The algorithm is as follows:

- An NDT-map from the current (unknown) pose of the robot is captured and keypoint descriptors are computed accordingly.
- The global NDT-map is split into fragments, which will then be independently matched with the current local NDT-map; the RANSAC outlier detection is executed each time as well.
- Subsequently, a special normalized inlier ratio r is computed for all pairings (see Eq. 6) and evaluated afterwards.

$$r = \frac{|M|}{\max(|D_A|, |D_B|)} \tag{6}$$

Here, $|M|$ denotes the number of inliers after RANSAC has finished. $|D_A|$ and $|D_B|$ represent the unfiltered amounts of keypoints found in map A and B, respectively. The reasoning behind this equation is as follows: If both maps would fit perfectly together, they would have a lot of keypoints in common as well as many correct matches. Hence, the inlier set would constitute a large proportion of the maximally possible amount of matches. Maps with little correspondence, however, will have a rather small inlier set compared to $\max(|D_A|, |D_B|)$, and the ratio r will be low. The best fitting global NDT-fragment will be selected as the one with the highest value r that is larger than an absolute threshold α and larger than the second best score times β. This way, the algorithm can find out if the local NDT-map has a clear favorite or whether all fragments fit equally well. If no distinct winner can be found, the robot should rotate a bit and try it again.

VII. Results

A. Comparison with State-of-the-Art

In order to provide objective reference regarding registration quality and performance of the proposed IRON registration algorithm, we used publicly available datasets for this benchmark [19]. In particular we used the datasets *"fr2/pioneer_slam"* and *"fr2/pioneer_slam2"*, as they both show a realistic robot drive through a scene with a variety of different shapes and objects and sufficient overlap between successive depth images (taken with a Kinect depth camera). Considering that accurate ground truth poses for the depth sensor are also given, the datasets are perfectly suited for a comparison of different registration methods.
In preparation, all depth images were converted into NDT-maps and point cloud files (see Fig. 7) and associated with their corresponding ground truth sensor poses.

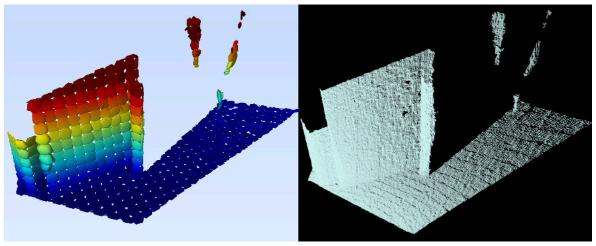


Fig. 7: NDT-map and point cloud, both extracted from the same depth image.

The registration benchmark was implemented as follows: Each NDT-map/point cloud is registered with a map which is t steps ahead of time. The parameter t directly influences the overlap that is available for matching. Therefore, the strongest overlap as well as the best registration scores are to be expected if t equals 1. This way, every map would be matched with its direct temporal successor.
One of both registration partners is now randomly displaced up to $4\,m$ away from its ground truth position and assigned a random orientation as well. The registration algorithm, which is currently tested, tries to realign both maps and the resulting transform is compared with the original ground truth pose of the displaced map. We defined an alignment to be successful, if the final translational error was below $5\,cm$ and the rotational error around each axis below $5°$. This process is then repeated 9 times for each pair of NDT-maps/point clouds. The following methods have been compared:

- *Method A*: IRON with keypoint detection enabled.
- *Method B*: IRON without keypoint detection (all IRON-descriptors will be available for matching).
- *Method C*: Iterative Closest Point, since it is one of the most widely used algorithms for point set registration.
- *Method D*: FPFH without keypoint detection. According to [20], the *Fast Point Feature Histograms (FPFH)*-descriptor has a very good registration quality and a relatively fast processing speed.
- *Method E*: FPFH with ISS keypoint detection. The *Intrinsic Shape Signatures (ISS)*-keypoint detector yields keypoints with high repeatability and was the fastest in a comparative evaluation of 3D keypoint detectors [21].
- *Method F*: FPFH without keypoint detection. Applied to point clouds that were down-sampled to resemble the size of NDT-cells.

- *Method G*: 3DSC with ISS-keypoint detection. 3DSC was chosen, since it does not create a unique frame of reference [9], which is also the case for IRON.
- *Method H*: 3DSC without keypoint detection. Applied to point clouds that were down-sampled to resemble the size of NDT-cells.

Except for the proposed IRON-registration, all other matching techniques were implemented by means of the popular point cloud library [22], and all search operations within point clouds were performed via k-d-tree data structures. The parameters were tuned in order to get the best registration quality for this benchmark, they are as follows:

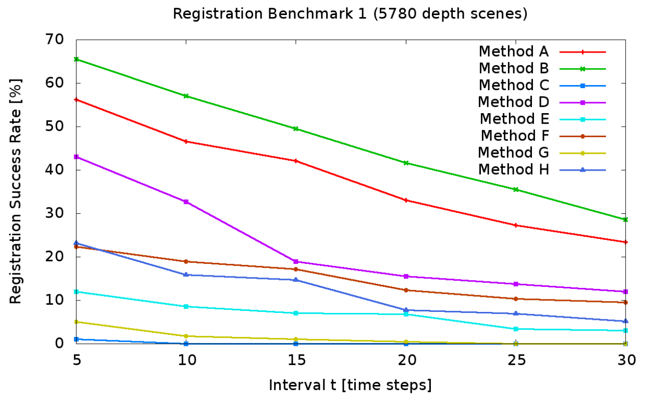
- Down-sampling of the input point clouds to contain no more than one point per cell within a $0.01m \times 0.01m \times 0.01m$ voxel grid (*C, D, E, G*), or $0.1\ m \times 0.1\ m \times 0.1\ m$ (*F, H*).
- Computation of surface normals for every point from both clouds, considering its 10 nearest neighbors.
- ISS-keypoint computation (radius for non maximum suppression: $0.04\ m$, radius for salient regions: $0.08\ m$) (*E, G*).
- Descriptor computation (*D, E, F, G, H*) (feature radius for *D, E, G*: $0.08\ m$, feature radius for *F, H*: $0.5\ m$)
- Descriptor matching is done via approximate nearest neighbor search inside the descriptor space.
- A RANSAC-based outlier rejection technique is applied (threshold: $0.05\ m$), the number of iterations for *E, G* is set to $1\,000$ and for *D, F, H* to $10\,000$, as the latter cases will potentially need to handle a larger proportion of outliers.
- The final transform for map alignment is computed from the remaining set of correspondences.

3DSC was not tested on resolution of $0.01\,m$ without keypoint detector, due to excessive computation time. ICP is simply applied to the down-sampled point clouds until convergence. Parameters for the IRON-registration algorithm were chosen in a similar fashion whenever possible; they are summarized hereafter:

- The NDT-cell size is set to $0.1\ m \times 0.1\ m \times 0.1\ m$.
- The radius of the spherical support region around a base cell is set to $\epsilon = 0.5\,m$ (Sec. III).
- Matrices $\mathbf{A}$ and $\mathbf{S}$ each have 3 rows and 3 columns, therefore the descriptor $\mathbf{D}$ consists of $m \times 2n = 18$ float elements (Sec. IV).
- The RANSAC-based outlier detection stage performs $1\,000$ iterations and the inlier threshold is set to $0.05\ m$.
- For *Method A* the entropy threshold for classifying keypoints is set to 0.6, for *Method B* it is 0.0, hence, all NDT-cells are considered keypoints and will be used for descriptor computation and matching accordingly (Sec. III).

Interestingly, the selection of the IRON feature radius as well as the sizes of $\mathbf{A}$ and $\mathbf{S}$ (within sensible limits), had only minor impact on registration quality. We found the following parameters to provide an especially beneficial trade-off between registration accuracy and processing time: ($m = 3$, $n = 3$, $\epsilon = 0.5\,m/0.6\,m$), ($m = 6$, $n = 6$, $\epsilon = 0.5\,m/0.6\,m$). Nevertheless, there is a wide range of parameters for IRON that give good results.

In Fig. 8, benchmark scores are now plotted depending on parameter t, with $\Delta t = 1$ corresponding to about $50\,ms$ delay between consecutive depth images from the given dataset. The larger t, the less overlap on average was available for matching. The vertical axis shows the percentage of successful map alignments that were achieved using a certain registration method.



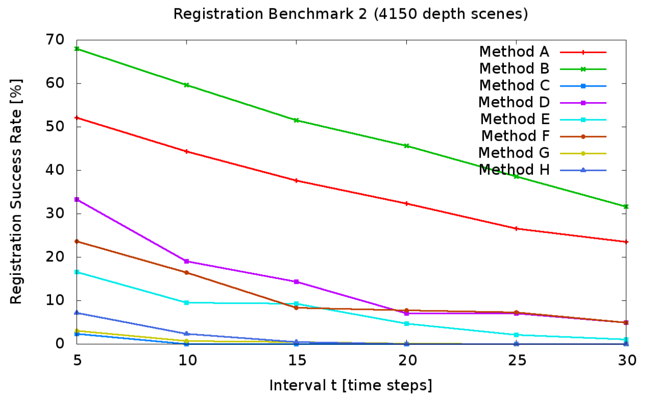


Fig. 8: Registration success rates in percent. High values correspond to a high amount of successful registrations. The best results (Method A and B) in both benchmark datasets were achieved using the IRON-registration algorithm.

For each method, $99\,300$ registrations were executed. The proposed IRON-registration algorithm achieved a success rate of more than 65% in both datasets when t was 5, and it exhibited good registration performance when the available map overlap was gradually reduced. It should be noted that most of the maps for which the IRON-registration failed, showed the ground or other flat surfaces, making it impossible to extract unique descriptors. When the IRON-keypoint detection was enabled, map matching was more than three times faster (Sec. VII-C), but registration quality was only slightly impaired. This shows that the detected NDT-interest points captured most of the obtainable information from the scene. Registration quality was worst for ICP, which can be attributed to the severe displacement between map pairs. The tested descriptor-based alignment techniques, however, were not susceptible to an initial offset or rotation and, hence, are better suited for the matching of depth data from Kinect sensors with a small viewing range.

B. *Further registration examples with high difficulty*

Obviously, less NDT-map overlap leads to fewer matches and potentially more outliers. To visualize registration performance in borderline cases, three NDT-map pairs with strikingly small overlap were registered, yet the IRON algorithm achieved correct alignment for all of them (see Fig. 9).

C. *Registration Speed*

The whole algorithmic design was kept as simple as possible and directed towards speed. Besides the assessment of registration quality in section VII-A, the average processing time of all methods was measured. This includes *complete memory allocation and complete initialization from scratch* for every single map registration procedure. All tests were done on *a single core* of an Intel i7-4770 CPU. See the following table (Tab. I) for average matching times of a single map-pair and the corresponding amount of registrations that could be performed within one second. Each algorithm executed 99 300 registrations for this evaluation using the benchmark datasets discussed in section VII-A.

TABLE I: Average registration times for all evaluated methods. IRON was by far the fasted of the tested algorithms.

Algorithm	*One registration[ms]*	*Matches per second*$[s^{-1}]$
A) IRON w. kp.	**13**	**76.9**
B) IRON no kp.	**45**	**22.2**
C) ICP	2544	0.4
D) FPFH no kp.	10558	0.1
E) FPFH w. kp.	5417	0.2
F) FPFH 0.1 m.	226	4.4
G) 3DSC w. kp	4661	0.2
H) 3DSC 0.1 m	2173	0.5

D. *Accuracy Estimation of the IRON-MCL*

As this work is primarily built around the IRON-registration itself, an exhaustive evaluation of the proposed localization algorithms would be beyond the scope of this paper. However, to give an estimate of the localization accuracy to expect, we used an NDT-SLAM-framework to create a global reference NDT-map of a large living room and to obtain the robot trajectory as perceived by SLAM. This trajectory is consistent with the created reference map and can therefore be used to evaluate the MCL algorithm. Figure 10 now shows the position error as Euclidean distance between the SLAM-trajectory and the IRON-MCL prediction at equal time steps.

E. *Estimated Performance of the One-Shot Localization*

In order to confirm proper functioning of the one-shot localization, a large living room was mapped and used as reference. Twentyfive random NDT-maps captured within the same scene (but from different locations) were then used for global localization (see Fig. 12). It was also noted whether the algorithm itself accepted a certain map as positive match (Tab. II); parameters were set to: $\alpha = 0.12, \beta = 2.0$. When the same 25 samples were tried to be fit into a reference map from an unknown living room, the algorithm automatically

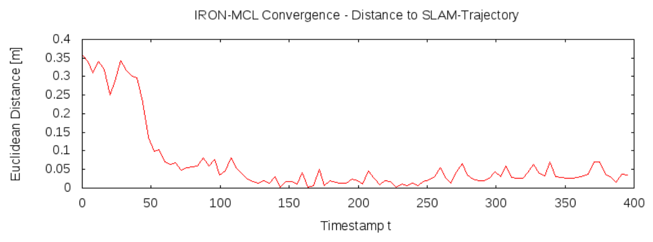


Fig. 10: Position error between SLAM-trajectory, and IRON-MCL-trajectory based on sensor recordings from a drive through a living area. A single time step corresponds to about 2 cm driven distance.

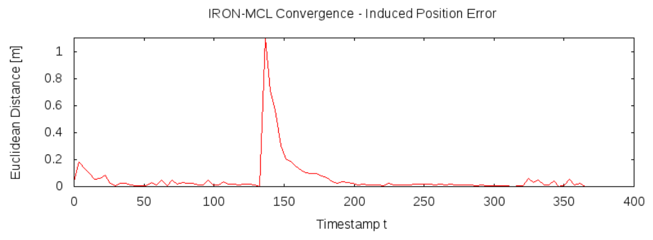


Fig. 11: This time, the robot was *manually* displaced 1 m at time step 140 and had to recover from this induced localization error.

rejected every single local map, in spite of the similarity between both environments.

TABLE II: Evaluation of the NDT-one-shot localization. In a large living area, twentyfive randomly captured NDT-maps were localized inside the global reference NDT-map.

Successful matches	16
Matches discarded autonomously	9
False positive matches	0

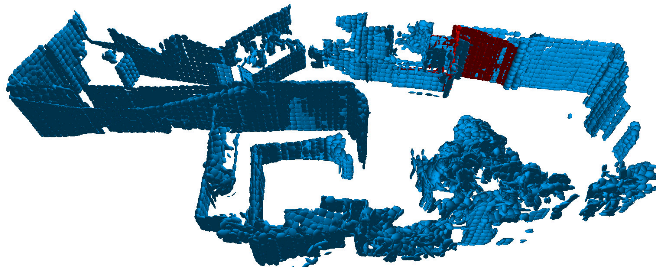


Fig. 12: A small NDT-map (red) was seamlessly aligned with the global reference (blue) using the proposed one-shot localization algorithm.

VIII. Conclusions and Future Development

This paper introduced the IRON interest point detector and the IRON descriptor and illustrated how they can be used to align NDT-maps with high accuracy and speed. Contrary to *Iterative Closest Point*-based approaches, the initial offset between the maps had no effect on registration quality, as was confirmed in section VII-A. Moreover, using publicly available depth-datasets we compared IRON to state-of-the-art methods for point cloud registration and found its success rate to be constantly higher than the success rates of all other tested algorithms, even when the available map overlap for matching was severely reduced. We attribute this result mainly to the fact that IRON descriptors always have

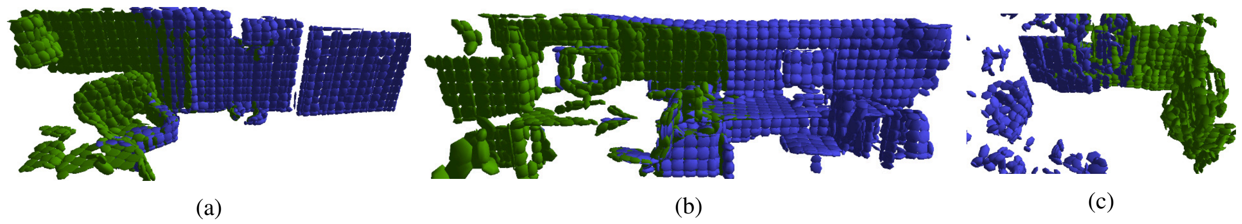


Fig. 9: Accurately registered NDT-map pairs, in spite of very little overlap.

the same appearance, independently of the sign of surface normal vectors (Sec. III). The other tested algorithms build local reference frames, and this makes a normal disambiguation step mandatory, which must be consistent among different maps. Additionally, due to the spatio-temporal low-pass filtering that happens when point values are converted into NDT-cells of discrete size, IRON is very resistant to clutter.

Due to several optimizations and design considerations with an eye towards processing speed, an average map registration with complete memory allocation only took $13\,ms$ (Sec. VII-C). This makes IRON well suited for real-time applications. One reason is, that the resolution of NDT-maps is typically quite coarse, hence, few elements have to be processed. Furthermore, IRON reuses the keypoint data structure as part of the descriptor, and no local reference frames need to be computed during the description process.

Altogether, for the tested realistic depth datasets, which are composed of more than 9 900 depth images, the proposed IRON-registration algorithm showed superior registration quality and a processing speed, which is several orders of magnitude faster than current state-of-the-art point cloud registration algorithms.

Later in this work, it was shown how feature-based NDT-matching can be used for pose tracking, with an approximate position error below $5\,cm$, once the MCL-particles have converged (Sec. VII-D). Additionally, this method displayed good ability to recover from (induced) localization errors, as was shown in Fig. 11. In order to obtain the robot's pose inside a known map but without any prior knowledge of its actual position and orientation, a technique was elaborated that registers the current NDT-map from the robot's field of view with a global reference in one shot. In the following test, using NDT-data from two different living areas, its basic functionality was confirmed. In addition, it could be seen that the algorithm either returned an exact match or automatically discarded the current NDT-sample if no distinct counterpart could be found. Hence, we did not detect any false positives (Sec. VII-E).

Future work must be dedicated to comparing the proposed localization techniques with the current state-of-the-art in terms of localization accuracy and speed on different publicly available benchmark datasets. Also, IRON is going to be implemented in an NDT-SLAM-framework, which will be evaluated as well.

The C++ source code of the IRON keypoint detector and the IRON descriptor, together with the complete registration algorithm can be downloaded at: `http://www.tu-ilmenau.de/neurob/data-sets-code/`.

References

[1] D. Fox, "Adapting the Sample Size in Particle Filters Through KLD-Sampling," *Int. J. of Robotics Research (IJRR 2003)*, vol. 22.

[2] M. Magnusson, A. Lilienthal, and T. Duckett, "Scan Registration for Autonomous Mining Vehicles Using 3D-NDT," *Journal of Field Robotics*, vol. 24, pp. 803 – 827, 2007.

[3] T. Stoyanov, M. Magnusson, H. Almqvist, and A. Lilienthal, "On the Accuracy of the 3D Normal Distributions Transform as a Tool for Spatial Representation," in *ICRA*, 2011, pp. 4080–4085.

[4] P. Jensfelt and H. Christensen, "Pose Tracking Using Laser Scanning and Minimalistic Environmental Models," *Robotics and Automation, IEEE Transactions on*, vol. 17, no. 2, pp. 138–147, Apr 2001.

[5] B. Steder, R. B. Rusu, K. Konolige, and W. Burgard, "NARF: 3D Range Image Features for Object Recognition," in *IROS*, 2010.

[6] T. Fiolka, J. Stueckler, D. A. Klein, D. Schulz, and S. Behnke, "SURE: Surface Entropy for Distinctive 3D Features," in *SC 2012*, vol. 7463, 2012, pp. 74–93.

[7] Y. Zhong, "Intrinsic Shape Signatures: A Shape Descriptor for 3D Object Recognition," in *ICCV 2009*, 2009, pp. 689 – 696.

[8] R. B. Rusu, N. Blodow, and M. Beetz, "Fast point feature histograms (fpfh) for 3d registration," in *Proc. ICRA*, 2009, pp. 1848–1853.

[9] A. Frome, D. Huber, R. Kolluri, T. Bulow, and J. Malik, "Recognizing Objects in Range Data Using Regional Point Descriptors," in *ECCV*, 2004.

[10] S. Salti, F. Tombari, and L. Stefano, "SHOT: Unique signatures of histograms for surface and texture description," *CVIU 2014*, vol. 125, pp. 251–264, 2014.

[11] T. Stoyanov, M. Magnusson, and A. Lilienthal, "Point set registration through minimization of the L2 distance between 3D-NDT models," in *ICRA*, 2012, pp. 5196–5201.

[12] J. Saarinen, H. Andreasson, T. Stoyanov, and A. J. Lilienthal, "Normal distributions transform Monte-Carlo localization (NDT-MCL)," in *IROS*, 2013, pp. 382–389.

[13] H. Gross et al., "Further Progress Towards a Home Robot Companion for People with Mild Cognitive Impairment," in *SMC*, 2012, pp. 637–644.

[14] H. Gross et al., "Robot Companion for Domestic Health Assistance: Implementation, Test and Case Study under Everyday Conditions in Private Apartments," in *IROS*, 2015.

[15] C. E. Shannon, "A Mathematical Theory of Communication," *The Bell System Technical Journal*, vol. 27, pp. 379–423, 623–656, 1948.

[16] M. Cummins and P. Newman, "Invited Applications Paper FAB-MAP: Appearance-Based Place Recognition and Mapping using a Learned Visual Vocabulary Model," in *ICML*, 2010.

[17] F. Dellaert, D. Fox, W. Burgard, and S. Thrun, "Monte Carlo Localization for Mobile Robots," in *ICRA*, 1999, pp. 1322–1328.

[18] E. Einhorn and H. Gross, "Generic NDT Mapping in Dynamic Environments and its Application for Lifelong SLAM," *Rob. Auton. Systems*, 2014.

[19] J. Sturm, N. Engelhard, F. Endres, W. Burgard, and D. Cremers, "A benchmark for the evaluation of rgb-d slam systems," in *IROS*, 2012.

[20] H. Kim and A. Hilton, "Evaluation of 3d feature descriptors for multi-modal data registration," in *3DV*, 2013, pp. 119 – 126.

[21] S. Filipe and L. Alexandre, "A comparative evaluation of 3d keypoint detectors," in *ConfTele*, vol. 1, 2013, pp. 145–148.

[22] R. B. Rusu and S. Cousins, "3D is here: Point Cloud Library (PCL)," in *ICRA*, 2011, pp. 1 – 4.